

A Differentiable Bayesian Anomaly Detection Framework for Robust SALT3 Parameter Estimation Using JAX

Sam Leeney

EUCAIFCON 2025

June 17, 2025

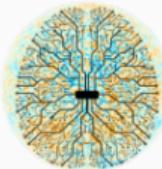
With: Will Handley, Harry Bevins, Eloy de Lera Acedo



UNIVERSITY OF
CAMBRIDGE



REACH



Outline

Bayesian anomaly detection

Bayesian anomaly detection for Ia Supernovae

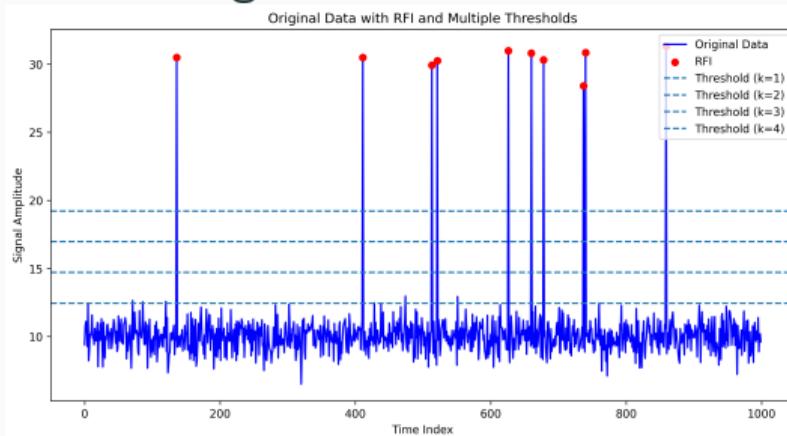
Fitting SALT models with JAX-bandflux

Fitting SNIa

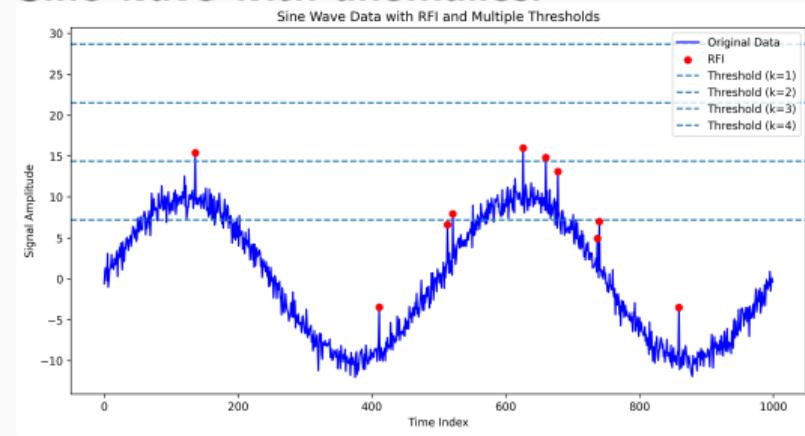
Bayesian anomaly detection

Over simplified example of anomaly detection method (thresholding)

Constant signal with anomalies:



Sine wave with anomalies:



- Traditional methods are generally not model aware.
- Anomalies are typically sought either before or after typical fitting process.

Model anomalies in via a method that is:

We want a method that is...

- Model aware
- Works simultaneously with model fitting
- Not binary, ie encodes 'belief' datum are anomalous

Bayes' Theorem

Bayes' Theorem:

$$P(\theta|\mathcal{D}) = \frac{P(\mathcal{D}|\theta)P(\theta)}{P(\mathcal{D})} \quad (1)$$

Where:

- $P(\theta|\mathcal{D})$: Posterior (updated belief)
- $P(\mathcal{D}|\theta)$: Likelihood (model prediction)
- $P(\theta)$: Prior (initial belief)
- $P(\mathcal{D})$: Evidence (model comparison)

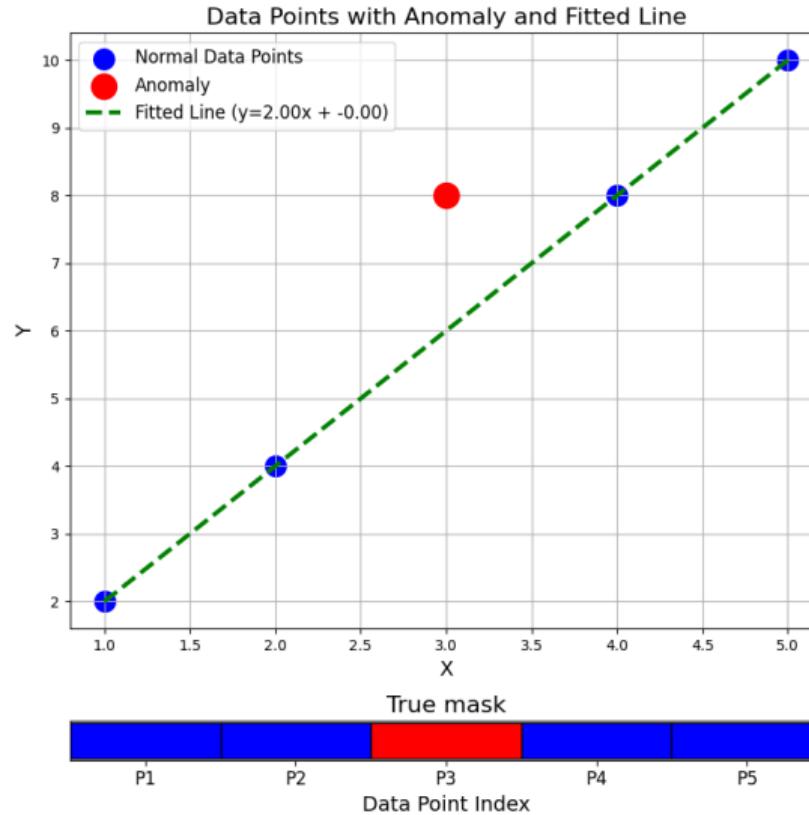
The Challenge:

We need some way of incorporating anomalous data points natively into our likelihood.

Standard likelihoods cannot account for this.

Define anomaly mask ε

$$\varepsilon_i = \begin{cases} 0 & : \text{expected} \\ 1 & : \text{anomalous,} \end{cases} \quad (2)$$



Ascribe Bernoulli prior to ε

$$P(\varepsilon_i) = p^{\varepsilon_i} (1 - p)^{(1 - \varepsilon_i)}. \quad (3)$$

- A Bernoulli prior assigns a probability p to a binary variable being 1 (anomalous) and $1 - p$ to it being 0 (expected).

Piecewise likelihood with ε

The likelihood function before marginalizing over ϵ is given by:

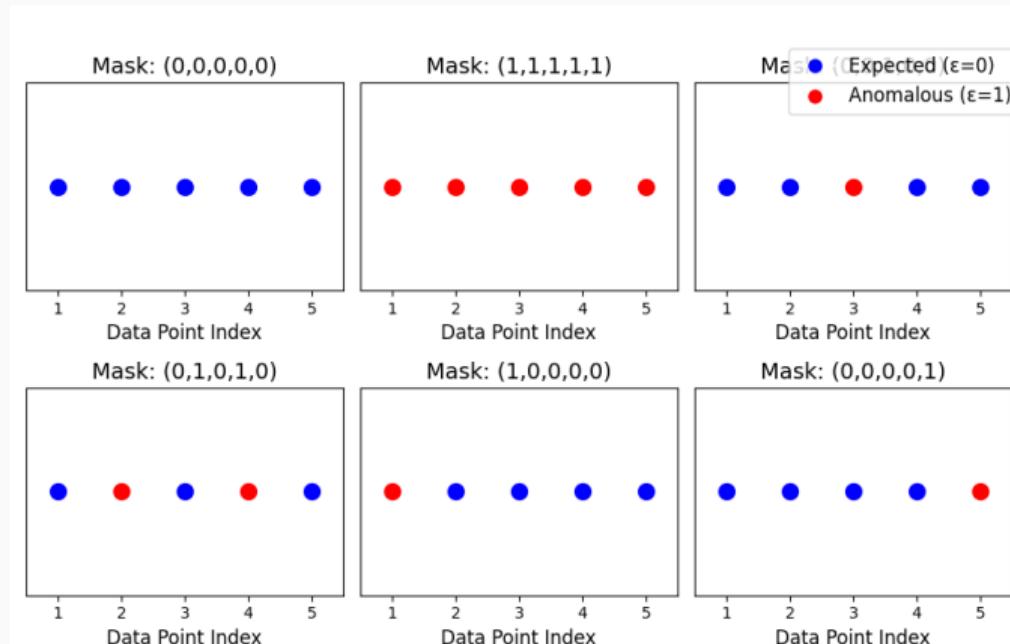
$$P(\vec{D}, \vec{\epsilon} | \theta) = \prod_{i=1}^N (L_i(\theta)(1-p))^{(1-\epsilon_i)} \left(\frac{p}{\Delta}\right)^{\epsilon_i}$$

Where:

- $L_i(\theta)$ is the likelihood of the i 'th data point D_i under the "expected" model.
- Δ is a constant related to the "anomalous" model.
- p is the prior probability that a data point is anomalous ($P(\epsilon_i = 1)$).
- ϵ_i is a binary variable: $\epsilon_i = 0$ for expected, $\epsilon_i = 1$ for anomalous.

Marginalise over epsilon

$$P(\mathcal{D}|\theta) = \sum_{\varepsilon \in \{0,1\}^N} P(\mathcal{D}, \varepsilon|\theta) \quad (4)$$



Likelihood After Marginalization

The likelihood function after marginalizing over ϵ is given by:

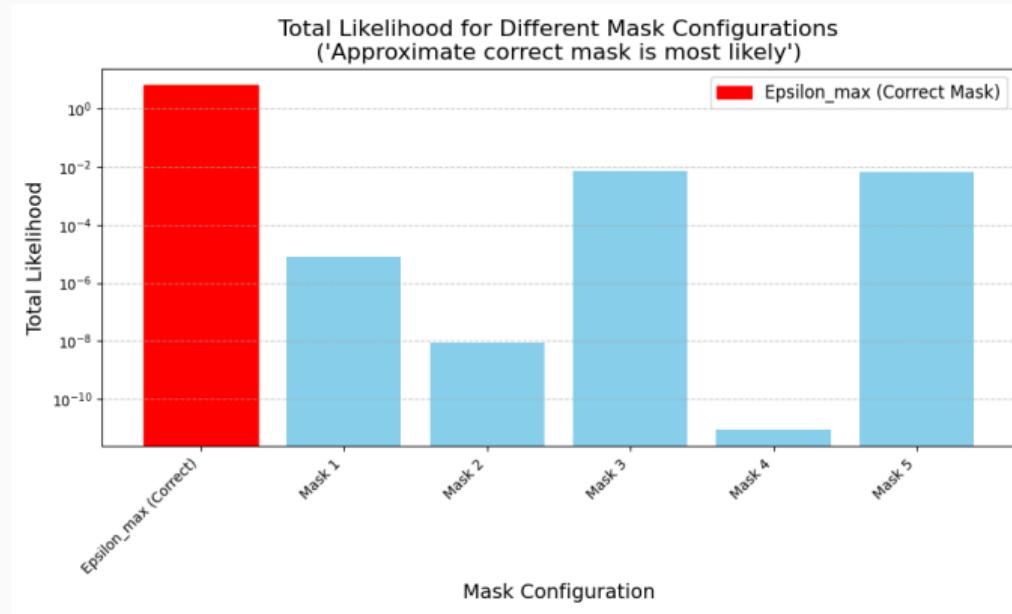
$$L(D|\theta) = \prod_{i=1}^N \left((1-p)L_i(\theta) + p\frac{1}{\Delta} \right)$$

Where:

- $D = \{D_1, D_2, \dots, D_N\}$ represents the dataset of N data points.
- θ represents the model parameters.
- $L_i(\theta)$ is the likelihood of the i -th data point D_i being "expected".
- p is the prior probability that a single data point is "anomalous".
- This is computationally impractical as mask scales 2^N .

Approximate correct mask is most likely

$$P(\mathcal{D}|\theta, \varepsilon_{\max}) \gg \max_j P(\mathcal{D}|\theta, \varepsilon^{(j)}), \quad (5)$$



Loglikelihood and Maximisation

e) Loglikelihood:

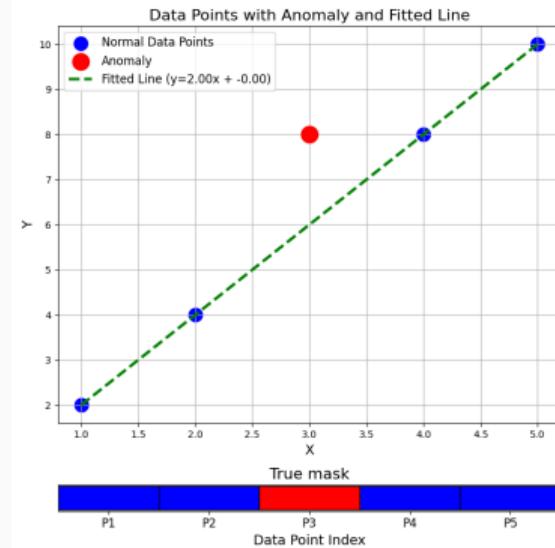
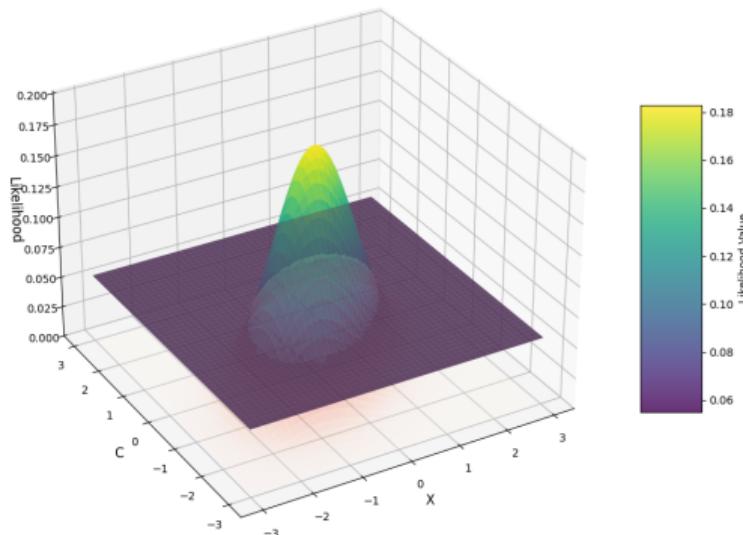
$$\begin{aligned}\log P(\mathcal{D}|\theta) = & \sum_i [\log \mathcal{L}_i + \log(1-p)] \varepsilon_i^{\max} \\ & + [\log p - \log \Delta] (1 - \varepsilon_i^{\max})\end{aligned}\tag{6}$$

f) Find the mask ε^{\max} that maximises the likelihood:

$$\log P(\mathcal{D}|\theta) = \begin{cases} \log \mathcal{L}_i + \log(1-p), & \text{if } \log \mathcal{L}_i + \log(1-p) > \log p - \log \Delta \\ \log p - \log \Delta, & \text{otherwise} \end{cases}\tag{7}$$

We are imposing a 'floor' on our likelihood

2D Gaussian Likelihood with a Flat Floor at p



Connection to the Logit Function

The condition for selecting between expected and anomalous data:

$$\log \mathcal{L}_i + \log(1 - p) > \log p - \log \Delta \quad (8)$$

Rearranging:

$$\log \mathcal{L}_i + \log \Delta > \log p - \log(1 - p) = \log \left(\frac{p}{1 - p} \right) \quad (9)$$

The right-hand side is the **logit function**:

$$\text{logit}(p) = \log \left(\frac{p}{1 - p} \right) \quad (10)$$

- Logit is the inverse of the sigmoid function
- Maps probability $p \in (0, 1)$ to $(-\infty, +\infty)$
- Common activation function in machine learning for binary classification
- Naturally encodes our degree of belief in each datum's integrity

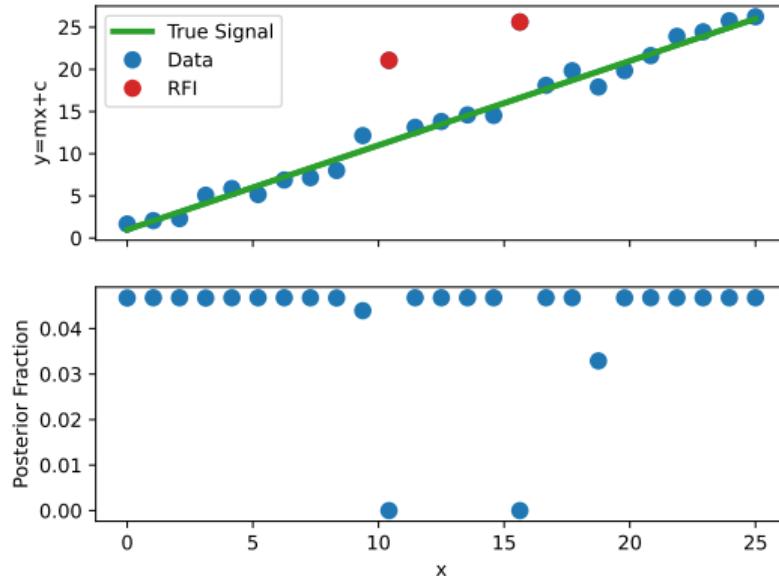
Derivation Summary

1. **Define anomaly mask:** $\varepsilon_i \in \{0, 1\}$ for each data point
2. **Bernoulli prior:** $P(\varepsilon_i) = p^{\varepsilon_i} (1 - p)^{(1 - \varepsilon_i)}$
3. **Piecewise likelihood:** $P(\vec{D}, \vec{\varepsilon} | \theta) = \prod_{i=1}^N (L_i(\theta)(1 - p))^{(1 - \varepsilon_i)} \left(\frac{p}{\Delta}\right)^{\varepsilon_i}$
4. **Marginalize over ε :** $P(\mathcal{D} | \theta) = \sum_{\varepsilon \in \{0, 1\}^N} P(\mathcal{D}, \varepsilon | \theta)$
5. **Dominant mask approximation:** $P(\mathcal{D} | \theta, \varepsilon_{\max}) \gg \max_j P(\mathcal{D} | \theta, \varepsilon^{(j)})$
6. **Final loglikelihood**

$$\log P(\mathcal{D} | \theta) = \begin{cases} \log \mathcal{L}_i + \log(1 - p), & \text{if } \log \mathcal{L}_i + \log(1 - p) > \log p - \log \Delta \\ \log p - \log \Delta, & \text{otherwise} \end{cases}$$

Any questions on the derivation before we proceed?

Fit on a simple toy model



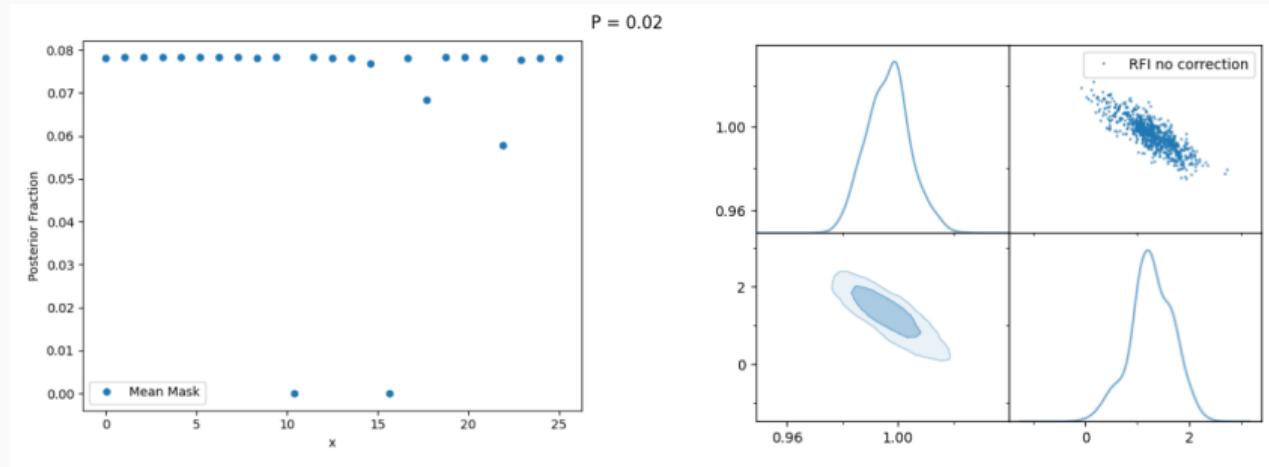
Posterior fraction:

$$f_i = P(\varepsilon_i = 1 | \mathcal{D}, \theta) \quad (11)$$

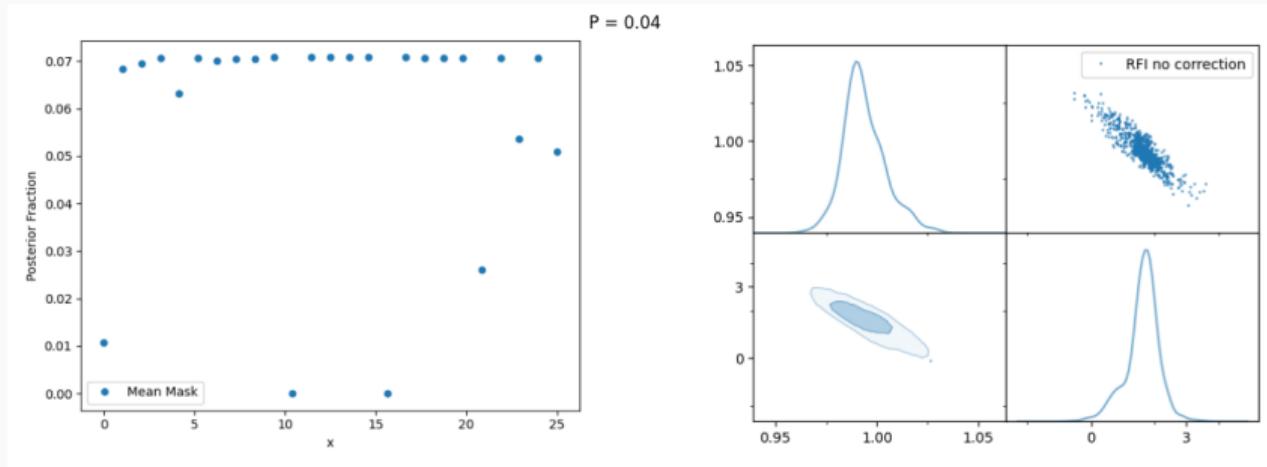
Posterior probability that data point i is anomalous.

Non-contaminated points have $f_i \approx 0.04$ (baseline level for 25 data points).

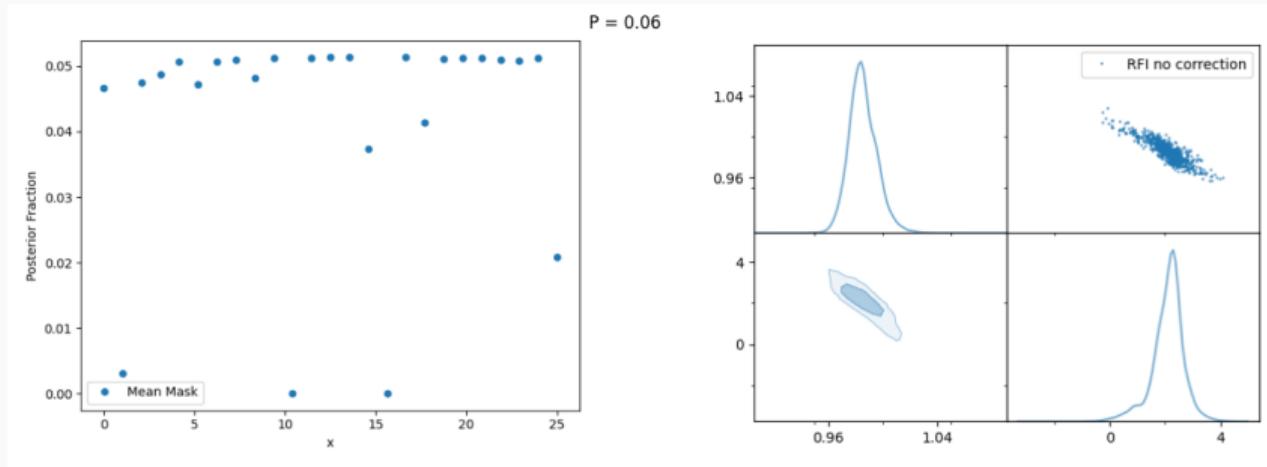
Varying p



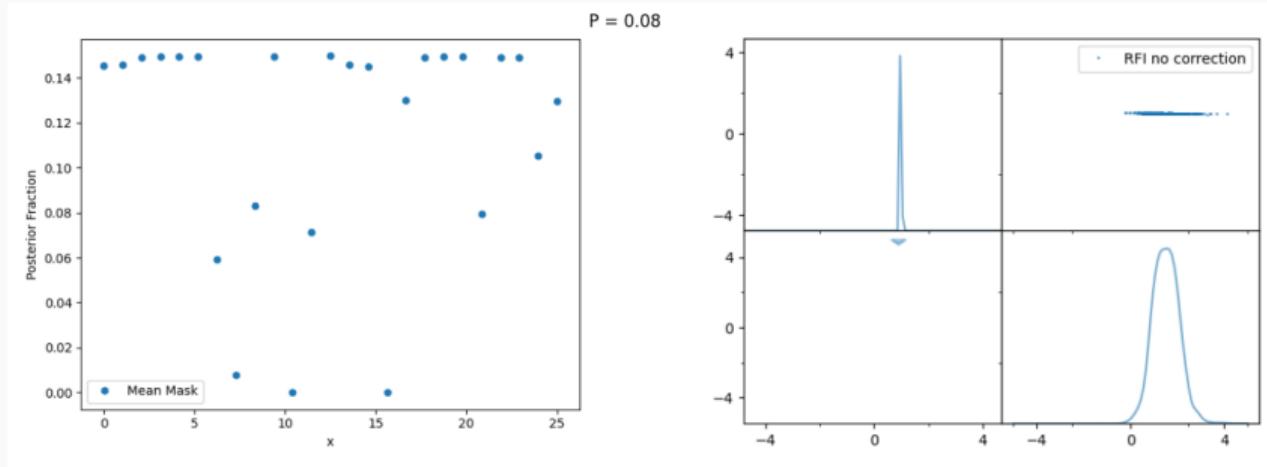
Varying p



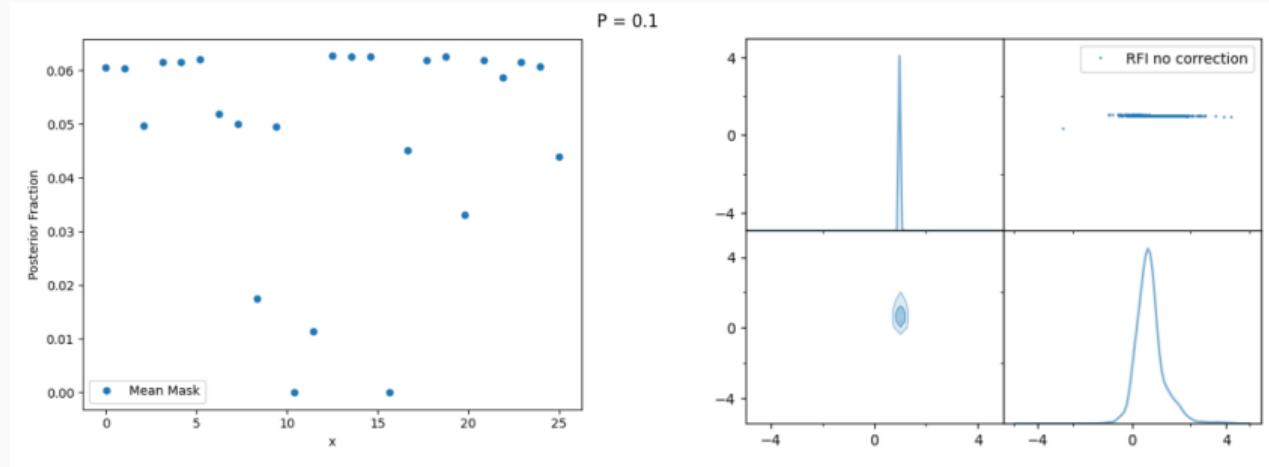
Varying p



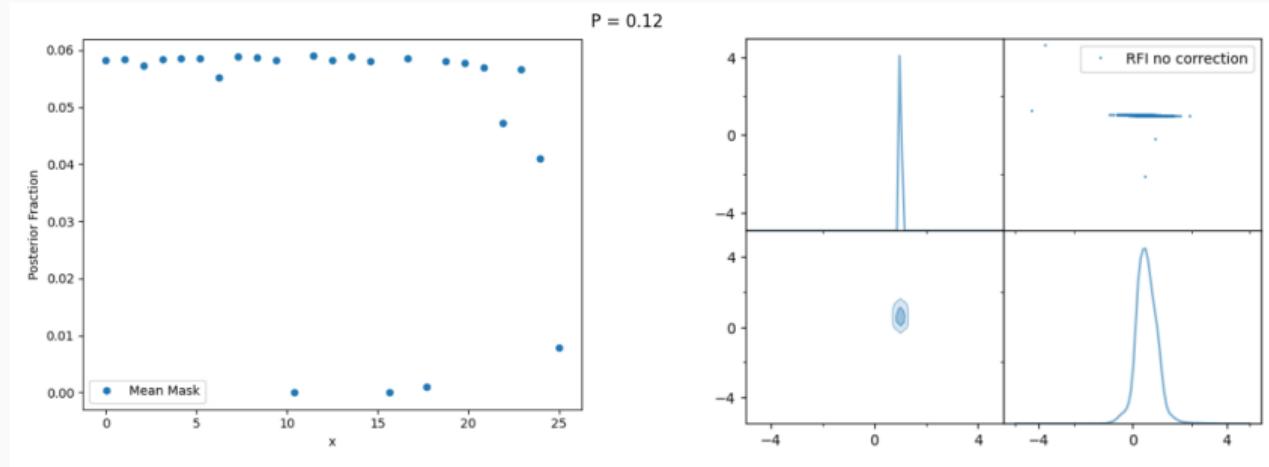
Varying p



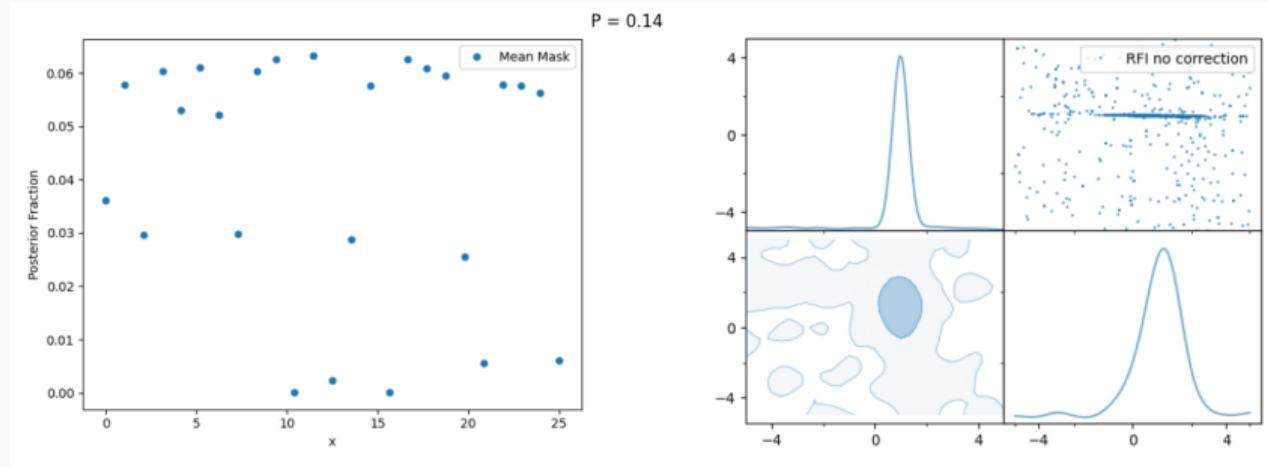
Varying p



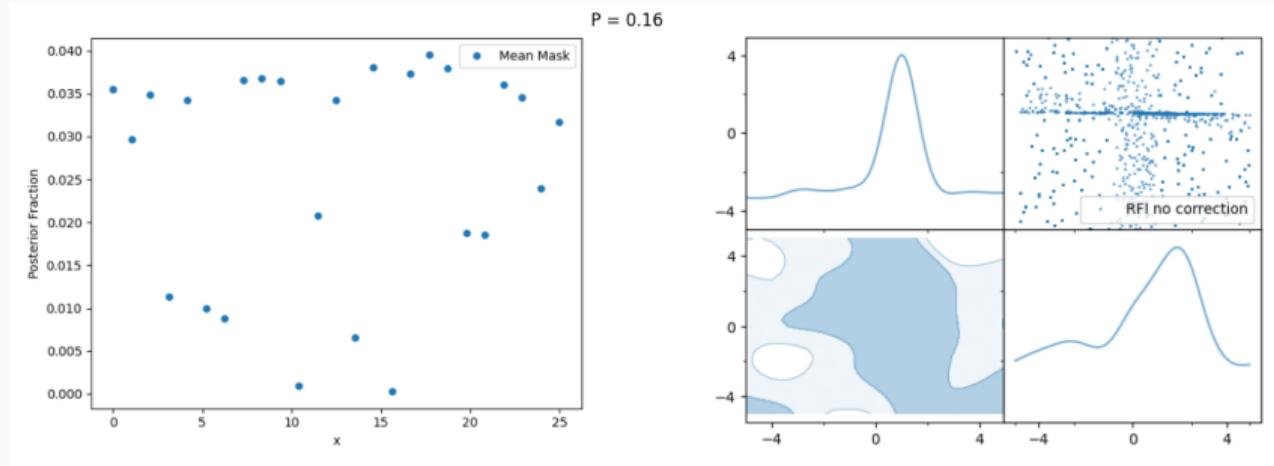
Varying p



Varying p

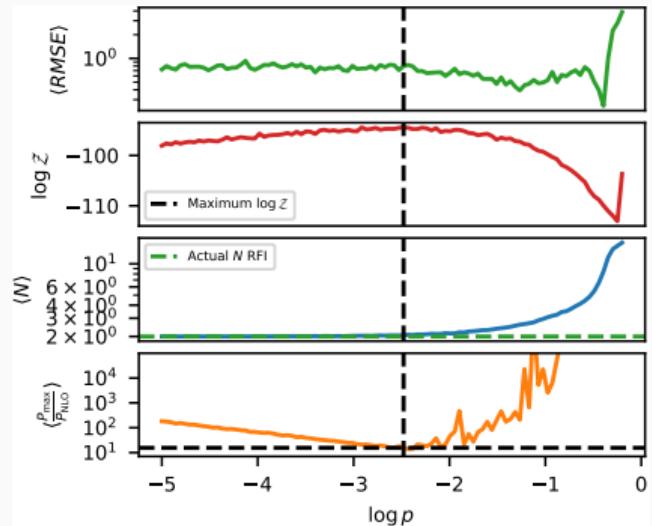


Varying p



Selection strategy for p .

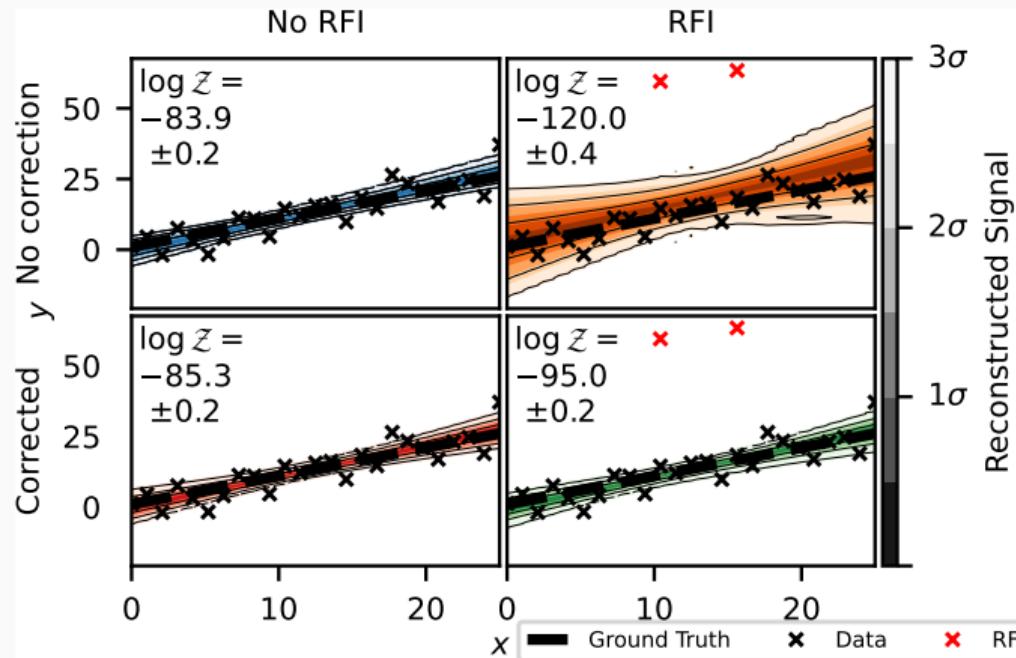
- ‘Select p such that the Bayesian evidence is maximised’



Fully automated anomaly detection

- Putting a prior on p , we can fit it dynamically as a free parameter.
- This fully automates the anomaly detection process.
- Must exclude $p = 0$.

Application to toy model



Implement with 2 lines of code

```
41
42 def likelihood(theta):
43     sig = theta[0]
44     logL = -(f_noise - window)**2/sig**2/2 - np.log(2*np.pi*sig**2)/2
45     return logL, []
46
34
47 def likelihood(theta):
48     sig = theta[0]
49     logL = -(f_noise - window)**2/sig**2/2 - np.log(2*np.pi*sig**2)/2 + np.log(1-p)
50     emax = logL > logP - np.log(delta)
51     logPmax = np.where(emax, logL, logP - np.log(delta)).sum()
52
53     return logPmax, []
54
```

Tutorial @ github.com/samleeney

Bayesian approach to radio frequency interference mitigation

S. A. K. Leeney^{✉,§}, W. J. Handley^{✉,§} and E. de Lera Acedo^{✉,§}

*The University of Cambridge, Astrophysics Group, Cavendish Laboratory,
J. J. Thomson Avenue, Cambridge, CB3 0HE, United Kingdom*



(Received 5 May 2023; accepted 29 August 2023; published 29 September 2023)

Interfering signals such as radio frequency interference from ubiquitous satellite constellations are becoming an endemic problem in fields involving physical observations of the electromagnetic spectrum. To address this we propose a novel data cleaning methodology. Contamination is simultaneously flagged and managed at the likelihood level. It is modeled in a Bayesian fashion through a piecewise likelihood that is constrained by a Bernoulli prior distribution. The techniques described in this paper can be implemented with just a few lines of code.

DOI: [10.1103/PhysRevD.108.062006](https://doi.org/10.1103/PhysRevD.108.062006)

arxiv: 2211.15448

Bayesian anomaly detection for Ia Supernovae

Supernovae Cosmology: Two Key Requirements

- **Standardisation Method:** Need a way to standardise and compare other supernovae
 - Corrects for intrinsic variations between SNe
 - Makes them useful as standard candles
- **Distance Anchor:** Need absolute distance calibration for some supernovae
 - Provides the absolute magnitude scale
 - Converts apparent magnitudes to distances

Both components are essential for cosmological measurements

Distance Anchor Methods

Distance Ladder:

- Cepheid variables
- Tip of Red Giant Branch (TRGB)
- Surface brightness fluctuations
- Direct geometric distances

Assumes Some Cosmology:

- Cosmic Microwave Background (CMB)
- Baryon Acoustic Oscillations (BAO)
- Early universe physics
- Sound horizon scale

H_0 tension: These methods give different results!

Standardising Supernovae

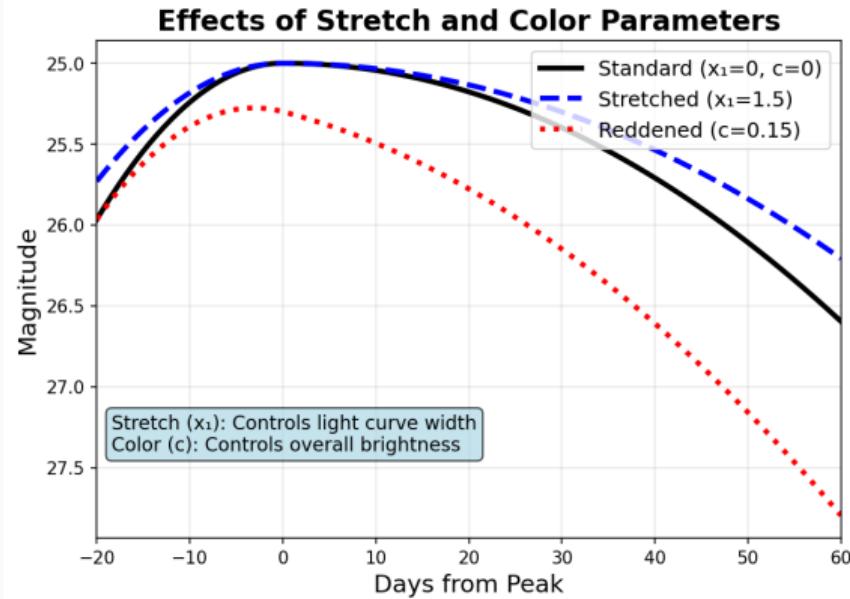
Key idea: If we can anchor one supernova's distance, we can standardise and compare it to all others

The Problem:

- Brighter SNe fade more slowly
- Redder SNe appear dimmer (dust)
- Prevents direct distance comparisons

The Solution:

- Measure stretch (x_1) and color (c)
- Apply SALT corrections
- Reveal common peak luminosity



Standardisation Methods

SALT Model (Spectral Adaptive Light curve Template):

- State-of-the-art standardisation method
- Parameters:
 - x_0 : amplitude/brightness
 - x_1 : stretch factor
 - c : color parameter
- Tripp formula:

$$\mu = m_B - M_B + \alpha x_1 - \beta c$$

Other Methods:

- SNooPy (Carnegie Supernova Project)
- MLCS2k2 (Multi-Color Light Curve Shape)
- SiFTO (SN Ia Fitter using Templates)
- BayeSN (Bayesian approach)

All methods aim to reveal:

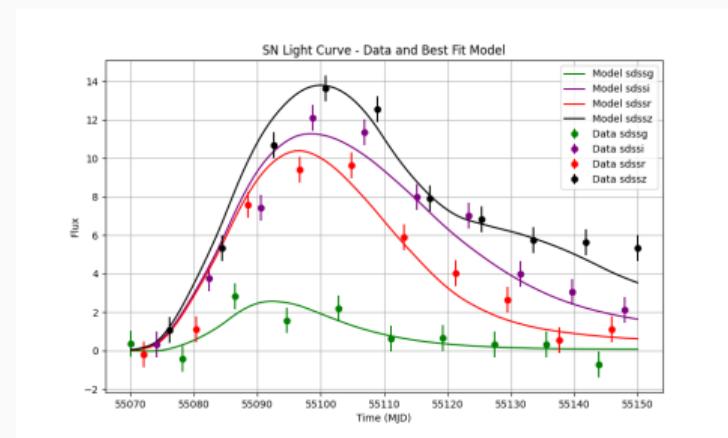
$$M_B \approx -19.3 \text{ at peak}$$

The SALT3 Model for Supernovae

- **The Inverse Problem:** We observe flux measurements $F_{\text{obs}}(\lambda, t)$ and want to estimate SALT parameters
- Given observations, infer:
 - x_0 : brightness scaling
 - x_1 : stretch (width)
 - c : colour
- Use SALT3 model as our forward model to relate parameters to observations

Forward model:

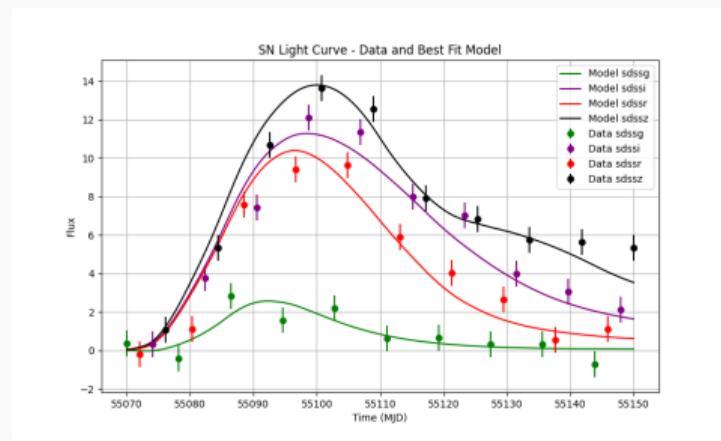
$$F(p, \lambda) = x_0 [M_0(p, \lambda) + x_1 M_1(p, \lambda) + \dots] \times \exp [c \times CL(\lambda)]$$



Bandflux Computation

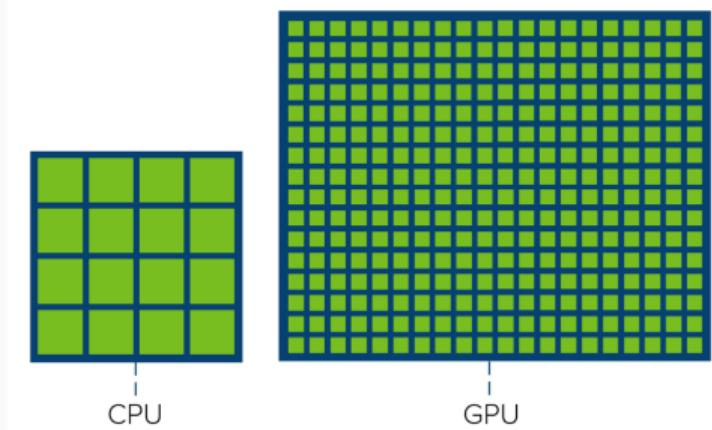
- **What we observe:** Bandflux measurements F_{obs} from telescope instruments
- **The challenge:** Estimate SALT parameters (x_0, x_1, c) from these integrated flux measurements
- **Forward model:** Predict bandflux from SALT parameters

$$F_{\text{model}} = \int_{\lambda_{\min}}^{\lambda_{\max}} F(\lambda; x_0, x_1, c) \cdot T(\lambda) \cdot \frac{\lambda}{hc} d\lambda$$



Fitting SALT models with JAX-bandflux

GPU vs CPU



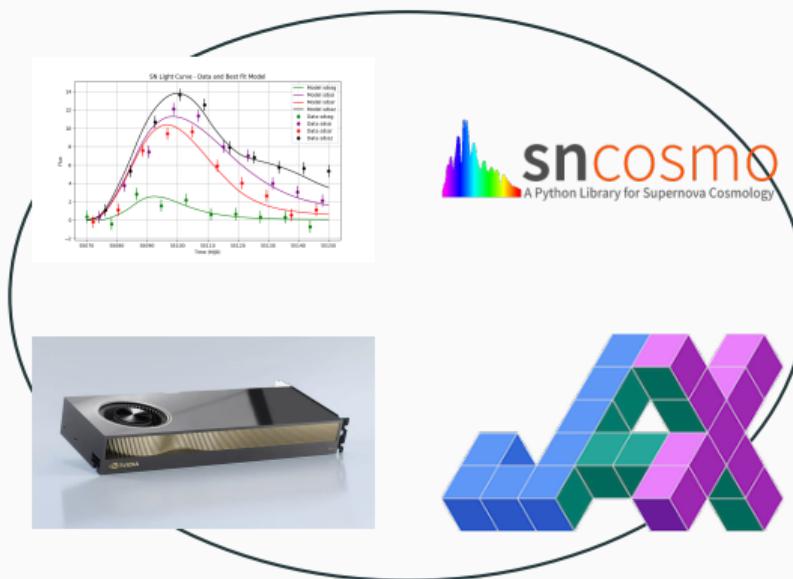
- Highly parallelisable
- Fast communication between operations
- GPU 'cores' do less, so no control flow

JAX numpy vs numpy

```
import numpy as np
x = np.arange(10)
y = np.zeros_like(x)
for i in range(len(x)):
    y[i] = x[i] * 2
```

```
import jax.numpy as jnp
from jax import vmap
x = jnp.arange(10)
def double(v):
    return v * 2
y = vmap(double)(x)
```

JAX-bandflux

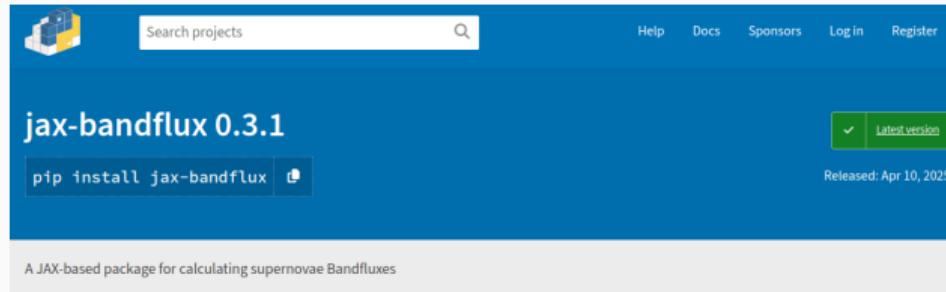


JAX-bandflux: A Tool for Supernovae Analysis

JAX-bandflux: differentiable supernovae SALT modelling
for cosmological analysis on GPUs

Samuel Alan Kossoff Leeney^{1, 2}

1 Astrophysics Group, Cavendish Laboratory, J. J. Thomson Avenue, Cambridge CB3 0HE, UK **2**
Kavli Institute for Cosmology, Madingley Road, Cambridge CB3 0HA, UK



JAX-based Nested Sampling Integration

NESTED SLICE SAMPLING

David Yallup*, Will Handley

Institute of Astronomy and Kavli Institute for Cosmology Cambridge
University of Cambridge
`{dy297, wh260}@cam.ac.uk`

Namu Kroupa

Department of Engineering and Cavendish Laboratory
University of Cambridge
`nk544@cam.ac.uk`

ABSTRACT

Nested Sampling is a powerful meta Bayesian inference algorithm known for its ability to estimate the marginal likelihood of a model and perform parameter inference, even for complex multimodal distributions. In this paper, we refine the formulation of Nested Sampling in the context of slice sampling, leading to a novel vectorized version of the algorithm that leverages GPU acceleration for improved efficiency in machine learning applications. We demonstrate that this vectorized Nested Slice Sampling algorithm can exploit parallelization opportunities to substantially reduce runtime while maintaining sampling accuracy. The performance of the approach is evaluated on a range of challenging benchmark problems, showing significant improvements in sampling efficiency and scalability to high dimensions. The proposed vectorized Nested Sampling algorithm opens up new possibilities for applying Nested Sampling to large-scale machine learning problems where efficient Bayesian inference is critical. We provide an open-source implementation of our method to facilitate adoption and reproducibility.

- BlackJAX integration
- GPU-accelerated sampling

Fitting SNIa

Standard Likelihood for Supernovae

For photometric observations with Gaussian uncertainties:

$$\mathcal{L}(\theta) = \prod_{i=1}^N \frac{1}{\sqrt{2\pi\sigma_i^2}} \exp\left(-\frac{(f_i^{\text{obs}} - f_i^{\text{model}}(\theta))^2}{2\sigma_i^2}\right)$$

- f_i^{obs} : observed fluxes
- $f_i^{\text{model}}(\theta)$: SALT model fluxes
- σ_i : flux uncertainties

Standard vs. Anomaly Detection Likelihoods

Standard Likelihood:

$$\log \mathcal{L}_{\text{std}} = -\frac{1}{2} \sum_i \left(\frac{f_i - m_i}{\sigma_i} \right)^2 - \frac{1}{2} \sum_i \log(2\pi\sigma_i^2) \quad (12)$$

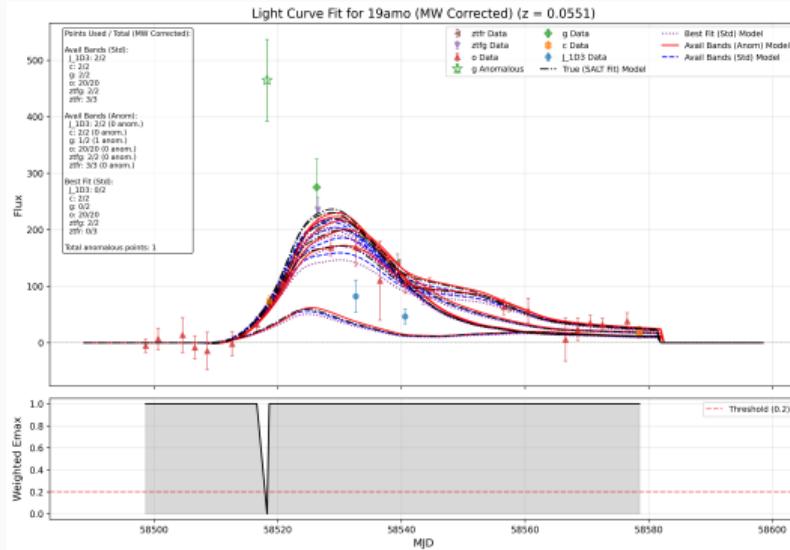
- f_i : Observed flux
- m_i : Model flux (SALT3)
- σ_i : Flux uncertainty

Anomaly Detection Likelihood:

$$\log \mathcal{L}_{\text{anom}} = \sum_i \begin{cases} \log \mathcal{L}_i + \log(1 - p), & e_i^{\max} \\ \log p - \log \Delta, & \text{else} \end{cases} \quad (13)$$

- $\log \mathcal{L}_i$: Point-wise likelihood
- p : Anomaly probability
- e_i^{\max} : Boolean for normal data
- Δ : Maximum flux range

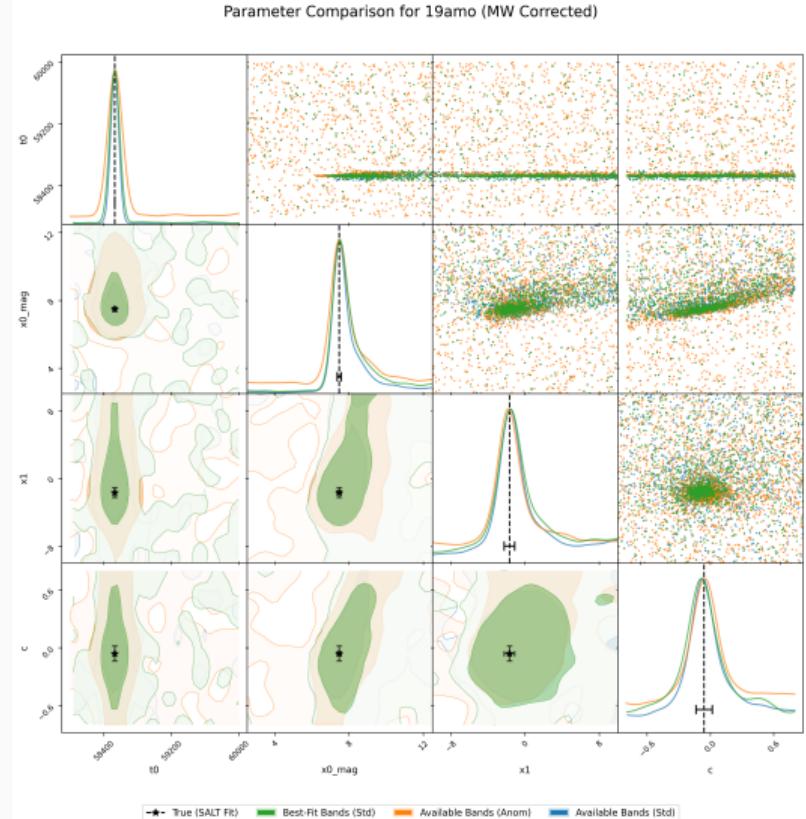
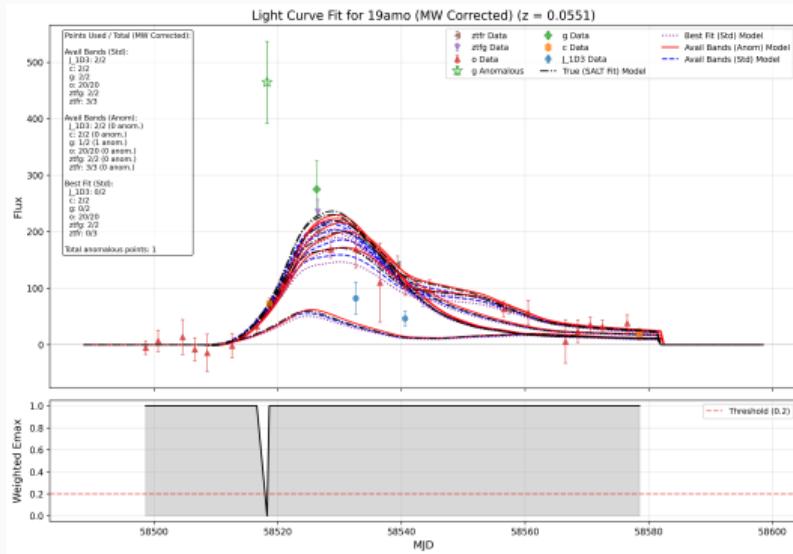
Automated Bandpass Selection Results



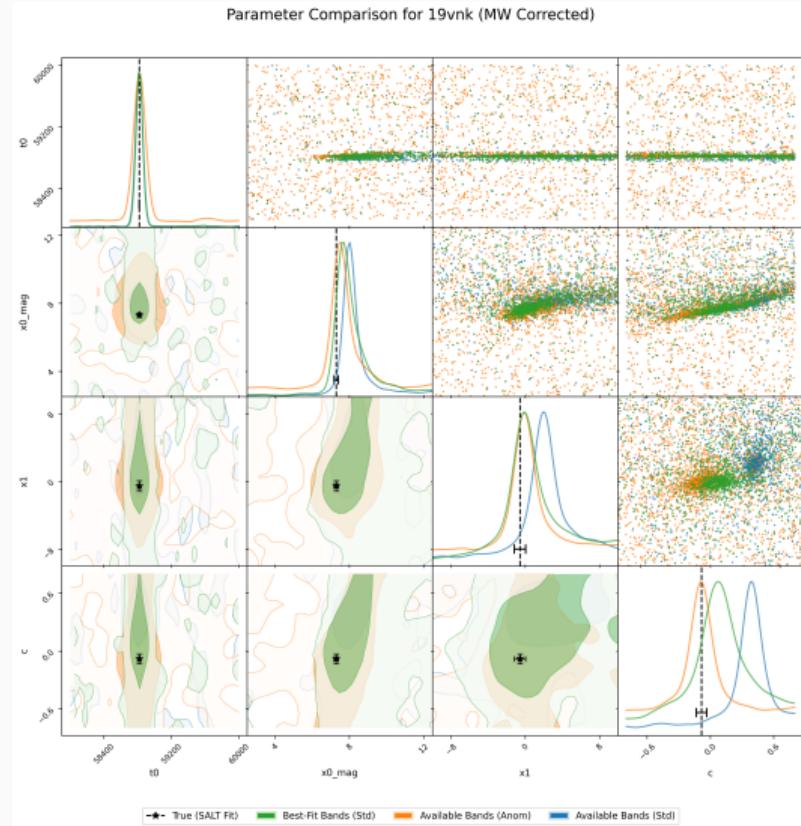
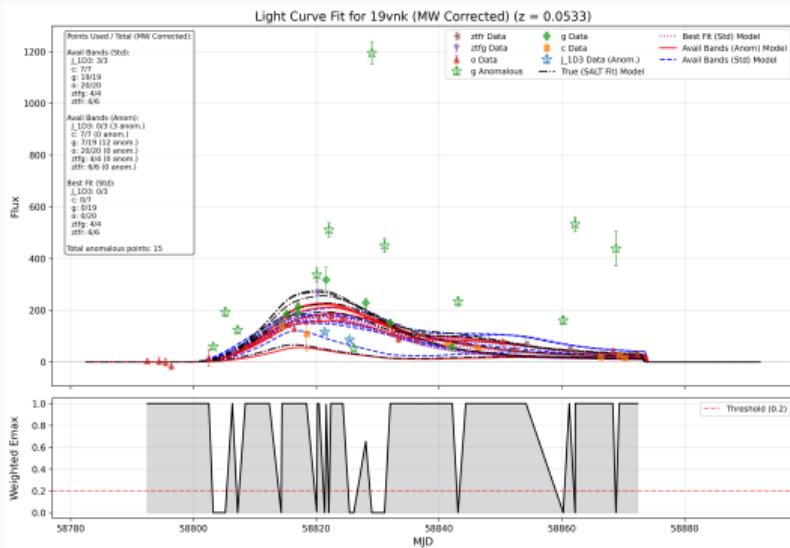
Our method demonstrates:

1. Standard flagging
2. Automatic filter selection
3. Data preservation

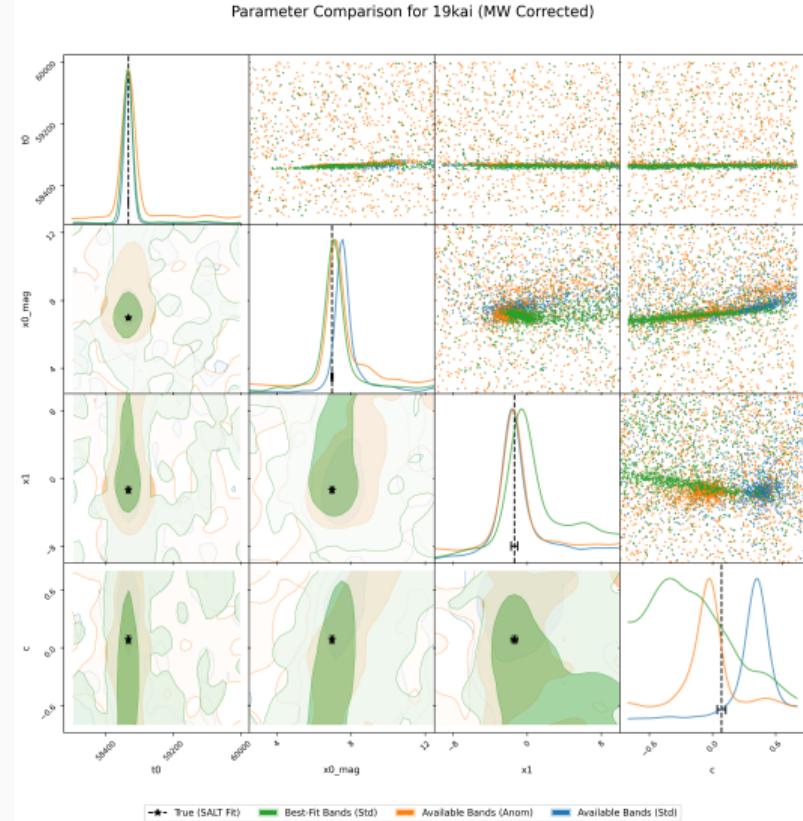
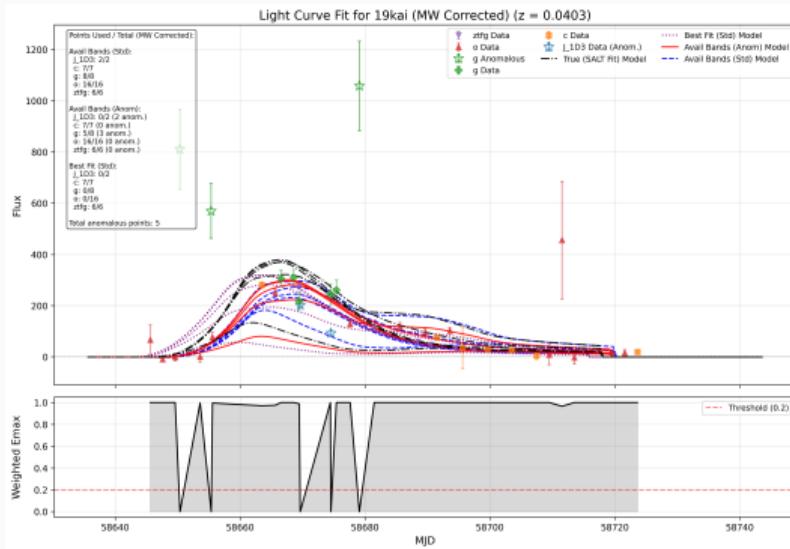
SN 19amo: Classic 'anomaly detection' example



SN 19vnk: Automatic filter removal



SN 19kai: Flagging while preserving some data



Key points

1. Standard flagging.
2. Automated filter selection.
3. Data preservation from previously discarded filters.

Next steps

1. Distance calibration

- Improve calibration accuracy using anomaly-aware methods
- Better handling of systematic uncertainties in distance measurements

2. H_0 measurements

- Apply to infer H_0
- More robust parameter estimation for cosmological inference

3. More datasets

- Extend to larger supernova surveys (LSST, Roman Space Telescope)
- Apply to different astronomical transients and surveys

Thanks for listening!

